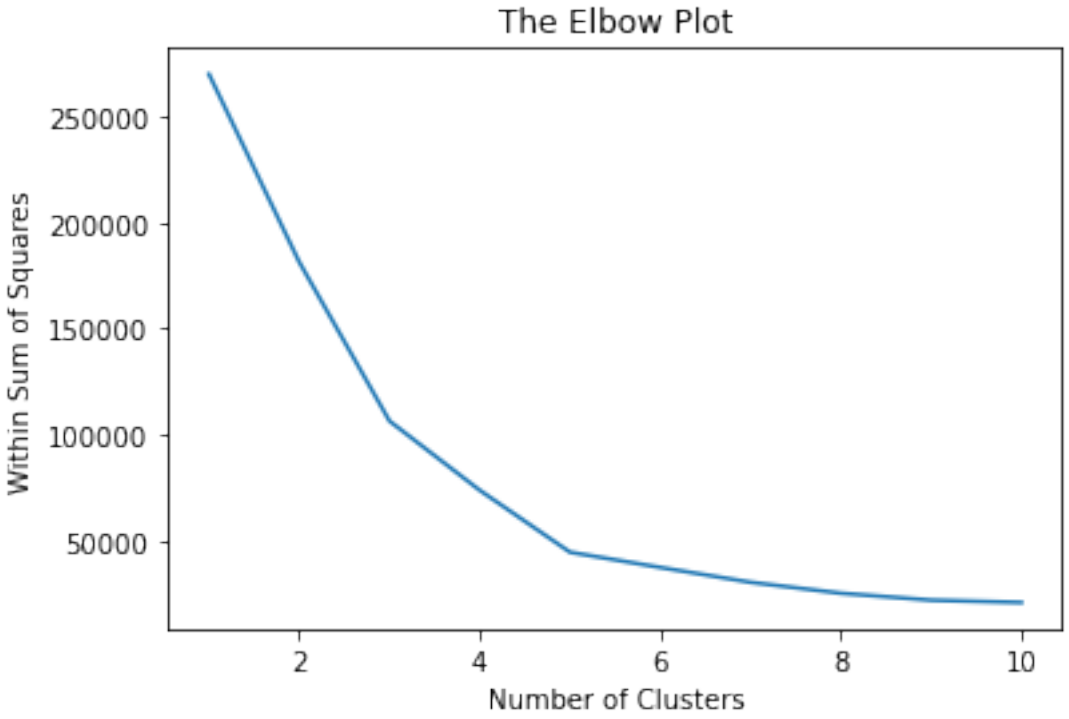
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

mydata = pd.read_csv("Mall_Customers.csv")
mydata.head()

   CustomerID  Gender  Age  Annual Income (k$)  Spending Score (1-100)
0           1    Male   19                  15                      39
1           2    Male   21                  15                      81
2           3  Female   20                  16                       6
3           4  Female   23                  16                      77
4           5  Female   31                  17                      40

iv = mydata[['Annual Income (k$)','Spending Score (1-100)']]

from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters = 2  , n_init =10 , random_state =0 )

kmeans.fit(iv)
kmeans.predict(iv)

array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1,
       1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
       0, 0])

# using elbow method to select number of clusters

## Code to find within sum of squares
wss= []
for i in range(1,11) :
    kmeans = KMeans(n_clusters = i, n_init =10 ,random_state =0 )
    kmeans.fit(iv)
    wss.append(kmeans.inertia_)  #Inertia: Sum of distances of samples
to their closest cluster center
    print (i, kmeans.inertia_)
```

```
#find the values where change slowing down .
```

```
C:\Users\Manish\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:881: UserWarning: KMeans is known to have a memory leak on
Windows with MKL, when there are less chunks than available threads.
You can avoid it by setting the environment variable
OMP_NUM_THREADS=1.
  warnings.warn(

1 269981.28000000014
2 181363.59595959607
3 106348.37306211119
4 73679.78903948837
5 44448.45544793369
6 37265.86520484345
7 30259.657207285458
8 25095.703209997544
9 21830.04197804944
10 20736.67993892413
```

```
## Plotting the Within Sum of Squares
plt.plot(range(1,11),wss)
plt.title("The Elbow Plot")
plt.xlabel("Number of Clusters")
plt.ylabel("Within Sum of Squares")
plt.show()
```



```
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters = 5 , n_init =10 , random_state =0 )
kmeans.fit_predict(iv)

# n_init : int, default: 10, Number of time the k-means algorithm will
be run with different centroid seeds.
```

```
array([4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4,
3,
       4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4,
1,
       4, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 0, 2, 1, 2, 0, 2, 0,
2,
       1, 2, 0, 2, 0, 2, 0, 2, 0, 2, 1, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0,
2,
       0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0,
2,
       0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0,
2,
       0, 2])
```

```
## Vizualizing the Clusters
iv['cluster']=kmeans.fit_predict(iv)
iv.head()
```

```
C:\Users\Manish\AppData\Local\Temp/ipykernel_9492/2304954149.py:2:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  iv['cluster']=kmeans.fit_predict(iv)
```

	Annual Income (k$)	Spending Score (1-100)	cluster
0	15	39	4
1	15	81	3
2	16	6	4
3	16	77	3
4	17	40	4

```
## Cluster Plot
plt.scatter(iv.loc[iv['cluster']==0,'Annual Income
(k$)'],iv.loc[iv['cluster']==0,'Spending Score (1-
```

```
100)'],s=100,c='red',label='Careful')
plt.scatter(iv.loc[iv['cluster']==1,'Annual Income
(k$)'],iv.loc[iv['cluster']==1,'Spending Score (1-
100)'],s=100,c='green',label='Standard')
plt.scatter(iv.loc[iv['cluster']==2,'Annual Income
(k$)'],iv.loc[iv['cluster']==2,'Spending Score (1-
100)'],s=100,c='blue',label='Target')
plt.scatter(iv.loc[iv['cluster']==3,'Annual Income
(k$)'],iv.loc[iv['cluster']==3,'Spending Score (1-
100)'],s=100,c='grey',label='Careless')
plt.scatter(iv.loc[iv['cluster']==4,'Annual Income
(k$)'],iv.loc[iv['cluster']==4,'Spending Score (1-
100)'],s=100,c='brown',label='Sensible')

#plt.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1]
, s=200,c='yellow',label='center')

plt.title("Results of K Means Clustering")
plt.xlabel("Annual Income")
plt.ylabel("Spending Score")
plt.legend()
plt.show()
```


```
iv[(iv['cluster']==4)][['Spending Score (1-100)','cluster']]
```

```
    Spending Score (1-100)  cluster
0                       39        4
```

2	6	4
4	40	4
6	6	4
8	3	4
10	14	4
12	15	4
14	13	4
16	35	4
18	29	4
20	35	4
22	5	4
24	14	4
26	32	4
28	31	4
30	4	4
32	4	4
34	14	4
36	17	4
38	26	4
40	35	4
42	36	4
44	28	4